US-PAT-NO: 6535912

DOCUMENT-IDENTIFIER: US 6535912 B1

TITLE: Method for creating and playing back a smart bookmark that automatically retrieves a requested Web page through a plurality of intermediate Web pages

---------- KWIC ---------

Abstract Text - ABTX (1):
Shortcuts to Web pages that require multiple steps to be retrieved are enabled by means of a smart bookmark. A smart bookmark is a stored sequence of browsing steps performed by a user, that have been recorded in a transparent manner and which can be automatically played and replayed later when the smart bookmark is accessed. When a user elects to create a smart bookmark, a Java recorder-player applet is invoked that starts the recording process. When the recording process is started and an initial URL is inputted by the user, the responsive Web page at that URL downloaded into the browser is modified to attach event handlers to each element in that page that is associated with actions that the user may take. Each user's click, link traversal to another URL, or input of values to those elements on a form submission are automatically recorded as part of the smart bookmark under creation. The resultant information at each step is recorded in a file. When the smart bookmark is later accessed, the recorder-player Java applet reads the file, and the sequence of recorded steps is played back, including information associated with all link traversals and form submissions. During playback, each intermediate Web page optionally can be displayed in the user's browser, or only the last page can be displayed. Further, during playback, transitions between successive steps can be automatic or can require an input from the user before a next step in the sequence is made.

Application Filing Date - AD (1):
19990831

Brief Summary Text - BSTX (10):
The HTML file is accessed when the user who created the smart bookmark, or another user, wants to retrieve the last or an intermediate Web page reached through the sequence of steps stored in the smart bookmark. When that HTML file is accessed, the recorder-player Java applet embedded in the file, or referenced in the file, is initiated to read in the bookmark data. The Web page corresponding to the URL of the first part of the smart bookmark is retrieved and optionally displayed on the user's browser. The next part of the smart bookmark file is then read. If the next part is a link traversal, a matching function based on the recorded link position, text and URL in the smart bookmark, along with corresponding properties of links present in the last Web page retrieved in the previous step during playback of the smart bookmark, is used to determine the best match for the link that should now be

traversed for the current step. This makes the play functionality more robust with respect to changes in the Web page structure that may have occurred between the time the smart bookmark was created and the current time when the smart bookmark is being played. The page corresponding the best matching URL is then retrieved and optionally displayed in the target window that may be specified in the bookmark. If the next part of the smart bookmark file is a form submission, a similar matching function is used to determine which form to submit. The value of each element in that form is determined from either the value stored in the bookmark data for that element, is decrypted from the value stored in the bookmark data, or is determined from a user's input in a pop-up window that requests input by the user of that element value. Certain elements retain the values set by the server generating that Web page. Once all the element values of the form submission have been determined, the form is automatically submitted to the Web server requesting it, and the resultant Web page is optionally displayed by the browser to the user. This resultant Web page may be the ultimate and final destination of the smart bookmark, or it itself may include link traversals, or include another form, the data associated with either being determined from the smart bookmark data and/or user input until, eventually, the Web page associated with the last step of the smart bookmark is displayed by the browser.

Brief Summary Text - BSTX (13):

The present invention can be implemented as a client-based operation where the smart bookmark recorder-player Java applet, individual recorder and player applets, or a single or separate recorder and player applications are embodied on the user's local machine together with a local smart bookmark database that stores one or a plurality of recorded smart bookmarks and which can be accessed by the user. Alternatively, the present invention can be implemented on Web server where the smart bookmark recorder-player Java applet, separate applets, or applications reside and which can be downloaded by a user for creating and playing a smart bookmark. When created, the bookmark can be stored locally on the user's own machine, or can be uploaded to that same Web server in the network for storage thereon, or to any other Web server, where it can also be stored. In order to replay a locally stored bookmark, if already downloaded, the recorder-player Java applet, separate player applet, or application, is invoked and the smart bookmark is played locally. If not already downloaded, the applet or application can be downloaded and the smart bookmark played. If the smart bookmark is stored on a Web server, it can be downloaded onto the user's local machine and played locally using a locally stored or downloaded recorder-player applet, player applet, or application. Alternatively, the smart bookmark can be played on the server where it is stored and the results sent back to the requesting user.

Detailed Description Text - DETX (9):

If, in the user input examined in step 106, the user neither hit the "save" button within the recorder window (determined at step 107), nor clicked on a link (determined at step 108), then, at step 110, it is determined that the user performed a form submission. The applet then stores in the smart bookmark the name of the form (if present), the DOM location of the form on the page, an action associated with the form (if any), and a method (GET or POST) associated with that form. Information is then determined for each element in the form. At step 111, the number (NUM) of elements within the form is determined. At

step 112, a form element number, I, is initialized at 1. At step 113, the current form element number, I, is compared with NUM. If I is less than NUM, at step 114, for element number I, the current value of that element is read in and the element name and DOM location of that element relative to the DOM location of the form is stored in the smart bookmark. If, at step 115, element number I is a hidden element that is specified by the server generating the form page rather than by user input, then the element is not stored in the smart bookmark. The element number I is then incremented by one at step 117, and information about the next element number within the form is determined. If element I is not determined to be a hidden element at step 115, then, at step 118, the user is prompted to assign a property that is associated with that element. The property can be: (1) store the element value in plain text; (2) encrypt and store the encrypted element value; or (3) don't store any element value and prompt the user when the smart bookmark is replayed. Thus, if the form requires an input of a user's identity or password for one or more elements, which the user does not want incorporated into the smart bookmark, in recording the smart bookmark, such elements are assigned the property of "store-in-encrypted-form" or "wait-for-user-input". The property selected by the user for such an element is then stored, at step 119, in the smart bookmark. If, at step 120, the property selected for current element I is "store in plain text", then, at step 121, the current element value (either entered by the user, or the existing default value) on the form is stored in the smart bookmark. The element number I is then incremented by one at step 117. If, at step 122, the property selected by the user is determined to be "wait-for-user input", then the current element value is not stored in the bookmark and the element number I is incremented by one at step 117. If the property selected for the current element I is neither "wait-for-user input" nor "store-in-plain-text", then, at step 123, the property of the current element value inputted by the user is "store-in-encrypted-text" and the current element value is encrypted (with a default encryption key, or an encryption key supplied by the user) and stored in the smart bookmark. Again, at step 117, the element number I is incremented by one.

Detailed Description Text - DETX (25):

The recording and playback functionalities could also be performed in the network. The recorded bookmark could then be downloaded into the client for later replay or could be stored in a central database. A smart bookmark, in that case would be made available to a user through a unique id generated by the database. Each smart bookmark would then be accessed through a URL that includes the database site and an encoding of the unique id. Such a URL can then be stored like any other URL in a conventional bookmark and sent as email. Any user could access that URL on a server in the network with which the smart bookmark database is associated. The smart bookmark would then be played on that site and the results sent back to the user who sent the smart bookmark reference to the server. Such a user would thus require no special software running on their client to be able to use the smart bookmark.